Haaga-Helia
ammattikorkeakoulu Oy

Tietovarastokuvausten takaisinmallinnus mallinnustyökalun avulla

Samuel Heino

Opinnäytetyö
Tietojenkäsittelyn
koulutusohjelma
2017

Haaga-Helia
University of Applied Sciences

Tiivistelmä

Tekijä

Samuel Heino

Koulutusohjelma

Tietojenkäsittely

Opinnäytetyön otsikko

Tietovarastokuvausten takaisinmallinnus mallinnustyökalun avulla

Sivu- ja liitesivumäärä

32

Opinnäytetyössä kuvataan, miten mallinnustyökalulla saadaan takaisinmallinnettua olemassa oleva tietovarasto sekä lisättyä tietovaraston tauluille ja sarakkeille määritelmät. Projekti on toteutettu toimeksiantona Keskinäinen eläkevakuutusyhtiö Varmalle. Sen tuloksena syntyi tietovaraston malli, joka sisältää taulujen ja sarakkeiden määritelmät sekä tietovaraston SQL luontilauseet. Opinnäytetyössä ei luotu uutta tietovarastoa.

Opinnäytetyö alkaa teoriaosuudella, jossa kerrotaan yleisesti operatiivisista tietokannoista ja tietovarastoista sekä selvitetään näiden eroavaisuuksia. Teoriaosuudessa käsitellään myös tietokantamallinnuksen ja normalisoinnin perusteita. Sen viimeisessä osuudessa esitellään takaisinmallinnusta.

Empiirinen osuus koostuu viidestä osiosta: nykytilanne, tavoite, käytetyt työkalut, työn toteutus ja lopputulos. Projekti käynnistettiin, koska kyseisestä tietovarastosta ei ole tällä hetkellä saatavissa tarkkaa tietomallikuvausta. Tavoitteena oli saada Varmalle valmis tietovaraston malli, joka pitää sisällään taulujen ja sarakkeiden määritykset.

Takaisinmallinnukseen käytettiin Idera Softwaren ER/Studio Data Architect -työkalua. Työ toteutettiin takaisinmallintamalla olemassa oleva tietovarasto ja lisäämällä määritykset 35 tauluun ja yli 550 sarakkeeseen. Lopputuloksena saatiin tietovarastosta HTML-raportti, joka sisälsi tietovaraston tietomallin, taulujen ja sarakkeiden määritelmät sekä tietovaraston SQL-luontilauseet.

Asiasanat

Tietovarasto, takaisinmallinnus, asiakkuudenhallinta, ER/Studio, dokumentaatio

Sisällys

1 Johdanto

Tämän opinnäytetyön perimmäisenä tarkoituksena on takaisinmallintaa olemassa olevan tietovaraston tietomalli ja -kuvaukset sekä selvittää tietovaraston taulujen ja sarakkeiden määritelmät. Opinnäytetyö tehdään toimeksiantona Keskinäinen Työeläkevakuutusyhtiö Varmalle, jonka asiakastietojen tietovarastosta on kyse. Opinnäytetyössä ei luoda uutta tietovarastoa, vaan keskitytään vanhan tietovaraston dokumentaation tuottamiseen helposti ylläpidettävään muotoon. Tavoitteena on luoda mallinnustyökalulla takaisinmallinnetun tietovaraston tietomalli sekä SQL-lauseet, joilla sarakkeiden määritelmät voidaan lisätä tietovarastoon.

Opinnäytetyön teoriaosuudessa tarkastellaan tietokantoja ja tietovarastoja yleiseltä kannalta. Otetaan myös selvää, mitä tehtäviä tietovarastoihin liittyy ja miksi tietovarastointia tehdään. Teoriassa käydään läpi tietokantamallinnuksen perusvaiheita sekä yleisimmät tietovarastomallit. Teorian lopuksi käsitellään myös hieman takaisinmallinnusta, jota tarvitaan empiirisen osan toteuttamisessa. Teoriaosuudessa on pääosin käytetty kahta suurta lähdettä: tietovarastoinnin isänä tunnetun Ralph Kimballin teosta sekä Suomen johtavaa tietovarastoinnin asiantuntijaa Ari Hovia. Näiden asiantuntijoiden teoksista saa luotettavimman ja täsmällisimmän tiedon.

Empiirisessä osassa käydään läpi tämän toimeksiannon työn tavoite ja tarve sekä käytetyt työkalut. Suurimmaksi osaksi opinnäytetyössä työkaluna käytetään Ideran mallinnusohjelmaa nimeltä ER/Studio. Työ itsessään jakaantuu kolmeen osioon: olemassa olevan tietovaraston takaisinmallinnukseen, taulujen ja sarakkeiden määrityksiin ja tietomallin html- raportin tekoon. Lopuksi käydään läpi työn tulokset.

1.1 Sanasto

Attribuutti
tietokannan taulun sarake

CRM (Customer Relationship Management)
asiakastietojen hallintajärjestelmä

DBMS (Database Management System)
Tietokone ohjelma, jolla hallinoidaan tietokantoja.

Dimensiomalli
Tietovarastomalli, jossa tietovaraston taulut muodostuvat dimensio- ja faktatauluista.

Entiteetti
tietokannan taulu

ETL (Extract, Transform, Load)
Tietovarastoinnin työvaihe, jossa lähteestä haetaan, muokataan ja ladataan data tietovarastoon.

Informatica
ETL-työkalu tietovarastolatauksien kehittämiseen ja hallinnointiin

Latauskartta (Mapping)
Pohjapiirros, jolla tietovarasto ohjelma ohjataan hakemaan lähteestä dataa, muokkaamaan se oikeaan muotoon ja lataamaan haluttuun kohteeseen.

Lumihiutalemalli (Snowflake schema)
Dimensiomalli, jossa tietovarasto on normalisoitu kolmanteen normaalimuotoon. Malli muistuttaa muodoltaan hieman lumihiutaletta.

ODBC (Open Database Connectivity)
Rajapinta, jolla tietokoneohjelma voidaan yhdistää lukemaan tietokantaa.

Pääavain (PK Primary Key)
tietokannan rivin päätunniste

Scripti

Tietokoneohjelma, joka on luotu käyttäen ohjelmointikieltä.

Skeema

Tietokannassa skeemalla tarkoitetaan omistajaa, jonka sisälle tietokannassa asetetaan taulut, näkymät, ym. Tietokannassa voi olla monta skeemaa ja jokaiseen voidaan määrittää omat käyttöoikeudet käyttäjille.

SQL (Structured Query Language)

Ohjelmointikieli, jolla hallinnoidaan tietokantoja.

Takaisinmallinnus (Reverse engineering)

Työtapa, jolla tutkitaan jo tehtyä, jotta saadaan sen pohjapiirros selville.

Tähtimalli (Star schema)

Dimensiomalli, jossa tietovarasto on denormalisoitu toiseen normaalimuotoon. Malli muistuttaa muodoltaan hieman tähteä.

Viiteavain

Sarake tai sarakeyhdistelmä, jolla voidaan osoittaa jonkun muun taulun rivi käyttäen sen pääavainta.

2 Tietokanta

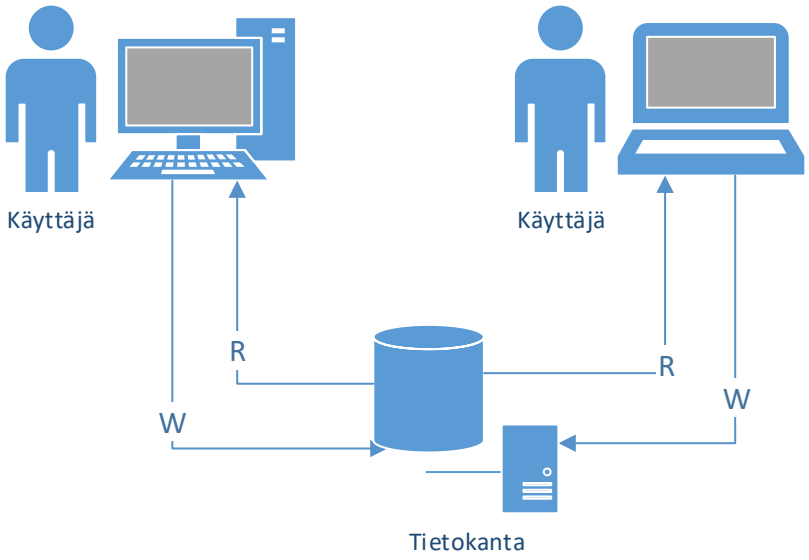
Tietokannat ovat tietokoneiden sovellusten, ohjelmistojen ja nettisivujen taustalla toimivia työkaluja, joihin tallennetaan järjestelmissä tarvittavaa tietoa nopeaa tiedonhakua ja -tallennusta varten (Merriam-Webster). Esimerkiksi tietokantoihin tallennetaan jonkin sovelluksen käyttäjätiedot mukaan lukien myös salasanat. Eli jos joskus olet unohtanut jonkin sovelluksen salasanan ja joudut painamaan ”unohditko salasanasi” painiketta, lähtee pyörimään prosessi, joka loppujen lopuksi tallentaa uuden salasanasi tietokantaan.

Tietokantatyyppejä on erilaisia: esimerkiksi operatiivisia ja analyyttisiä. Operatiiviset tietokannat on suunniteltu jatkuvaa muokkausta, hakua ja tietojen lisäämistä varten. Analyyttiset tietokannat on tarkoitettu tiedon analysointia varten. Kerron näistä tarkemmin seuraavissa kappaleissa.

2.1 Operatiivinen tietokanta

Operatiivisella tietokannalla tarkoitetaan tietokantaa, joka on yhteydessä sovellukseen reaaliaikaisesti. Nämä sovellukset ovat koko ajan käyttäjien käytössä ja heidän tekemät muutokset siirtyvät reaaliajassa tietokantaan. Operatiivinen tietokanta mahdollistaa sovellukseen nopeita tiedonmuutoksia, joita käyttäjä tarvitsee päivittäisessä käytössä. Tietokantaan kohdistuu paljon lyhyitä pistemäisiä tapahtumia: tiedon lukua ja päivitystä. Tämä mahdollistaa reaaliaikaisen tiedon saannin, mikä on erittäin tärkeää sovelluksen toimivuuden kannalta. (Hovi, Ylinen & Koistinen 2001, 45-46)
Esimerkiksi asiakashallintajärjestelmä, jota asiakaspalvelijat käyttävät, on suurella todennäköisyydellä yhteydessä tietokantaan. Näin asiakaspalvelijan tekemät muutokset on välittömästi muiden samaa sovellusta käyttävien nähtävillä.

Operatiivisen tietokannan luonteen takia saattaa syntyä ongelmia, joihin pitää suhtautua sopivalla vakavuudella. On erittäin tärkeä tarkkailla operatiivisen tietokannan kokoa, sillä sinne voidaan luoda koko ajan uutta tietoa, mistä johtuen tietokannan koko saattaa kasvaa nopeasti. Tällöin sovelluksen ylläpitäjän on tärkeä hankkia tehokkaat laitteet tietokannan taustalle mahdollistaakseen sen sujuvan käytön. Useilla yrityksillä onkin vaatimuksien kautta asetettuna aikarajat, kuinka kauan tietoa säilytetään tietokannoissa. Operatiiviseen tietokantaan tallennetaan harvoin tiedon historiaa, useimmiten pelkästään uusin tieto. Tätä varten yrityksillä on käytössä tietovarastoja.(GeekInterview, 2007)



Kuva 1. Esimerkkimalli tietokannan ja käyttäjien yhteyksistä (R Read eli lue ja W Write eli kirjoita)

Kuvassa 1 on esimerkki operatiivisen tietokannan toiminnasta. Molemmilla käyttäjillä on päätteellä olevan sovelluksen kautta yhteys tietokantaan. Kun he tekevät jotain uutta tai muuttavat vanhaa tietoa sovelluksessa, lähtee tästä muutoksesta tieto tietokantaan. Tämän jälkeen toinen käyttäjä voi lukea muuttuneen tiedon omalla sovelluksella, kun muutos on mennyt tietokantaan. Tietokantoja hallitaan SQL-kielellä (Structured Query Language). Tietokannan hallintaohjelmia on erilaisia ja jokaisen valmistajan sovellus eroaa hieman toisistaan, mutta pääpiirteittäin ohjelmointikieli on kaikissa sama (Troels 2011).

2.2 Relaatiotietokanta

Ari Hovi kertoo kirjassaan (2001, 45), että yleisin käytetty operatiivisen tietokannan tyyppi on relaatiotietokanta. Relaatiotietokanta on tietokanta, joka on rakennettu relaatiotietomallin perusteella. Relaatiomallissa tietokanta koostuu relaatioista eli tauluista, joiden sisällä on monikkoja eli rivejä, jotka muodostuvat attribuuteista eli sarakkeista. Relaatiotietokannan taulut näyttävät tiedot kaksidimensionaalisena taulukkona, samanlaisena kuin esimerkiksi Excel. Taulujen välisiin suhteisiin käytetään pää- ja viiteavaimia, jotta saadaan nopeasti yhdisteltyä tietoa eri osista tietokantaa. Näin saadaan data tallennettua helposti luettavaan ja haettavaan muotoon. (Oppel 2010, 13-14)

Taulukko 1. Esimerkki relaatiotietokannan taulusta, jossa on autojen tietoja.

Rivi_Id(PK)	Rek_nro	Merkki	Malli	Väri	Omistaja
1	AAA-111	Skoda	Octavia	Punainen	Pentti Perustaistelija
2	BBB-222	Opel	Astra	Vihreä	Erkki Esimerkki

Taulukossa 1 on kuvattu relaatiotietokannan taulu, joka sisältää autojen ja auton omistajien tietoja. Esimerkissä ensimmäinen sarake Rivi_id toimii taulun pääavaimena (PK eli primary key).

2.3 SQL- kieli

SQL eli Structured Query Language on tietokannoissa käytetty täsmäkieli, jolla ohjelmoidaan ja hallitaan tietokantoja. Sen kehittivät IBM:llä töissä olleet tietojenkäsittelyn ammattilaiset Donald D. Chamberlin ja Raymond F. Boyce (Chamberlin & Boyce 1974). SQL-kieli on koko tietokantahallinnoinnin ydin: sillä hoidetaan niin tietokannan rakentaminen kuin tiedon lataaminen tauluihin. SQL:ää käytetään ottamalla yhteys tietokantaan ja tekemällä sinne kyselyitä (query), joilla saadaan haluttu tieto näytölle. SQL ohjelmointikielen avulla voidaan myös luoda (CREATE), muuttaa (ALTER) ja poistaa (DROP) tauluja, sekä lukea (SELECT), lisätä (INSERT), muuttaa (UPDATE) ja poistaa (DELETE) tietoa tauluista (Chapple 2016).

2.4 Tietovarastointi

”Tieto on valtaa” on vanha sananlasku, joka pätee vielä tänäkin päivänä. Tieto on yksi yrityksen arvokkaimmista pääomista. Tätä omaisuutta on tärkeä osata analysoida oikein, jotta yrityksessä voitaisiin optimoida toimintoja. Tiedon analysoinnin avulla voidaan myös saada tietoja ongelmista yrityksen toiminnassa: mitä nopeammin ongelmat saadaan tietää, sitä nopeammin voidaan asioihin puuttua. Operatiiviset tietokannat eivät sovellu tämänlaiseen tiedon analysointiin, koska sovelluksia, joihin operatiiviset tietokannat ovat yhteydessä, ei ole rakennettu raportointimielessä. Tässä esiin astuvat tietovarastot. Tietovarastot on luotu juuri yrityksen erilaisten tietojen analysointiin ja säilyttämiseen. Näin myös vähennetään riippuvuutta operatiivisesta järjestelmästä, ja saadaan yrityksen tarvitsemat tiedot kootusti yhteen paikkaan. Tällöin operatiivisen järjestelmän vaihtaminen on helpompaa. (Hovi ym. 2001, 32-33)

Tietokaira-niminen yritys on saanut tiivistettyä tietovaraston pääperiaatteen yhdeksi lauseeksi: ”Tietovarastolla (Data Warehouse) tarkoitetaan yleensä isoa, monien lähdejärjestelmien tiedoista integroitua erillistä tietokantaa, johon säännöllisin väliajoin ladataan lähdejärjestelmistä uudet ja muuttuneet tiedot.”

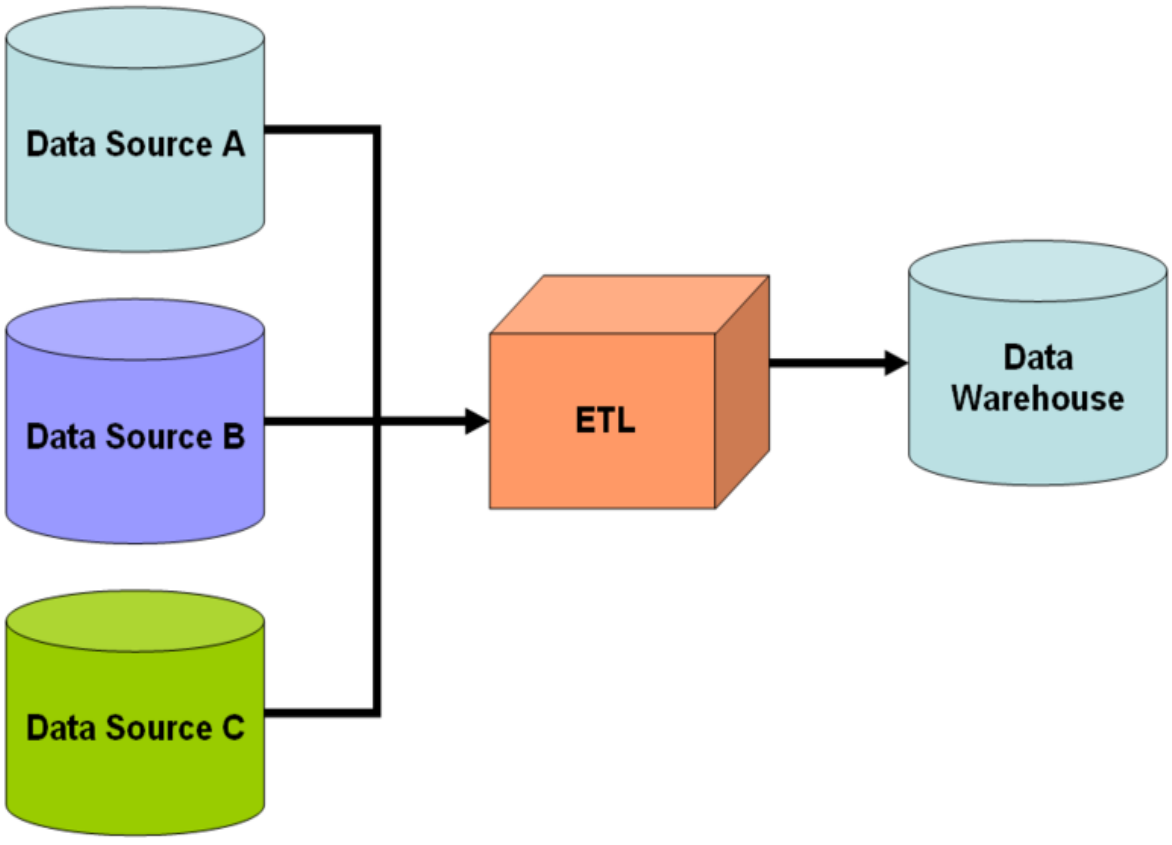
Tietovarastot ovat käytössä nykyään lähes kaikilla sovellusalueilla. Esimerkiksi ”clickstream analysis” -internetsivuilta kerääntyy käyttäjistä erittäin paljon tietoa: mitä käyttäjä on tutkinut, kauanko käyttäjä on ollut sivuilla ja minne käyttäjä siirtyi internetsivulta (Hovi ym. 2001, 33).

Toisin kuin operatiivinen tietokanta tietovarastot eivät sisällä täysin reaaliaikaista tietoa, eikä se myöskään ole niiden tarkoitus. Tietovarasto ei ole jatkuvassa yhteydessä mihinkään operatiiviseen sovellukseen, vaan niiden tarkoituksena on varastoida ja analysoida tietoa, jotka ladataan operatiivisista tietokannoista. Tietovaraston tietoja hyödynnetään erilaisissa raporteissa ja mittareissa: esimerkiksi, kun yrityksellä on operatiivisessa tietokannassa useampi taulu, siirretään tietovarastoon tarvittavat tiedot, jotta voidaan tuottaa tarpeeseen tarvittavat raportit. Tietovarastosta ei ole tarkoitus tehdä jatkuvasti kyselyjä, joten se voi sisältää erittäin paljon dataa. Operatiivinen tietokanta on käytössä jatkuvasti ja sieltä pitää pystyä nopeasti viemään sovellukseen tuoretta tietoa aina vaadittaessa. Tietovarastoissa on tarkoitus säilyttää tietoa pitkältä aikaväliltä, jotta operatiivisen tietokannan käytettävyys ei kärsisi. Tietovarastot sisältävät uuden tiedon lisäksi myös historiatietoa, jonka avulla voidaan luoda kattavia raportteja usealta eri aikaväliltä liiketoiminnan tueksi. (Hovi ym. 2001, 27-33)

Operatiivisissa järjestelmissä tietoa katsotaan näkökulmista, jotka sovelluskehittäjät ovat määritysten perusteella ohjelmoineet. Tietovarastoissa näkökulmat saattavat muuttua, joten on tärkeää, että tiedon laatu on kunnossa. Yleensä, kun tietoja katsotaan monesta eri näkökulmasta, löytyy virheitä, puutteita tai epäkohtia. Operatiivisella puolella nämä virheet, puutteet ja epäkohdat voivat olla vaikeasti havaittavissa, sillä yleensä operatiivisesta järjestelmän käyttäjät eivät ole kiinnostuneita tietojen tarkkuudesta tai oikeellisuudesta, kunhan väärät tiedot eivät vaikuta operatiiviseen toimintaan. Hovi esittää tilanteesta esimerkin: norjalaisessa sairaalassa käyttäjät huomasivat, että potilaan saa nopeasti kuitattua ulos, kun hänen käyntinsä syyksi kertoo kohdunseinämän rappeutuman. Tämä helpotti hankalan järjestelmän käyttöä, mutta ei palvele tiedon analysoijia tai raportointia. Tietovarastoissa esimerkin kaltaiset tiedon laadun ongelmat ja operatiivisen järjestelmän virhetiedot tulevat helposti esille. Tietovarastoissa tämä ei käy päinsä, vaan tiedon pitää olla aina oikein, muuten väärät tiedot vääristävät raportteja. Mikäli tiedon laatuun ei voi luottaa, tietovarastoa ei voi ottaa käyttöön. Näin tietovarastosta tulee hukkainvestointi, joka ei tuota yritykselle hyötyä.(Hovi ym. 2001, 35-37)

2.4.1 ETL-lataukset

Jotta erilaisten työkalujen olisi mahdollisimman helppo tuottaa tarvittavia tietoja tietovarastosta, on suositeltavaa, että tietovarastoissa tieto olisi jo siinä muodossa, missä sitä halutaan lukea. Tätä tarkoitusta varten tietovarastoihin tuotu tieto muokataan haluttuun muotoon prosessilla, jota kutsutaan ETL-lataukseksi. ETL on lyhenne sanoista Extract, Transform, Load. Latauksia hyödynnetään tietovarastoinnissa, kun uutta tietoa tuodaan tietovarastoon. ETL-lataukset ovat tietovaraston rakentamisen työläin vaihe, mutta oikein tehtyinä ne helpottavat tietovaraston käyttöä huomattavasti. Tietovarastoinnissa on erittäin tärkeää, että vain haluttu tieto siirretään tietovarastoon (Hovi ym. 2001, 77-78). ETL-lataukset mahdollistavat tämän. ETL-latauksissa on nimensä mukaisesti kolme vaihetta: ensimmäisessä vaiheessa lähdejärjestelmästä poimitaan tarvittavat tiedot, toisessa ladattu tieto muutetaan halutunlaiseksi ja kolmannessa muutettu tieto ladataan tietovarastoon. (Kimball, Ross, Thornthwaite, Mundy & Becker 2008, 127-128)



Kuva 2. Esimerkki perinteisestä ETL-latauksesta. Vasemmalta lähtien tieto liikkuu nuolten osoittamaan suuntaan tietolähteistä ja ETL-latausten kautta päätyy tietovarastoon (Wikipedia 2012)

Eri operatiivisissa tietokannoissa samat asiat saattavat olla koodattu eri menetelmin. Esimerkki: operatiivisessa tietokannassa A sukupuolet on merkitty M ja F tarkoittaen Male ja Female, toisessa tietokannassa B sukupuolet on merkitty Mies ja Nainen. Nämä tiedot pitää siis muuttaa yhtenäisiksi tietovarastoon ladattaessa. ETL-latauksessa voidaan muuttaa sukupuoli niin, että kaikki ne sarakkeet, jotka tulevat lähdejärjestelmästä A muodossa M saisivat arvon Mies ja F saisi arvon Nainen. (Hovi ym. 2001, 81). Tietovarastojen lähteinä voi myös käyttää muitakin kuin operatiivisia järjestelmiä (Hovi ym. 2001, 76). Esimerkiksi Suomen Postin internetsivuilta saa ladattua Excel-tiedoston Suomen postinumeroista ja niihin liittyvistä kunnista. Näin tietovarastoon voidaan suoraan ladata taulu, josta löytyy Suomen postinumerot ja niihin liittyvät sijainnit. Mikäli jostain jonkun muun lähteen tiedoista puuttuu kaupunki, voidaan postinumeron avulla etsiä tuosta taulusta vastaava kunta tai kaupunki.

ETL-latausten yhtenä tarkoituksena on myös tarkastaa, että lähteistä tuleva tieto on oikeanlaista (Hovi ym. 2001, 79-80). Tällä tarkoitetaan, että tiedon laatu on kunnossa, sillä lähdejärjestelmistä saattaa puuttua tietovarastojen kannalta olennaista tietoa. Esimerkiksi: operatiiviseen järjestelmään saatetaan olla koodattu etukäteen, että näytetään vain voimassa olevat asiakkaat. Operatiivisen järjestelmän taustalla pyörivässä tietokannassa saattaa kuitenkin löytyä vanhoja asiakkaita tai asiakkaita, joiden tiedot ovat puutteelliset. Näitä puutteellisia tietoja ei ladata tai asiakas merkitään jotenkin ennen tietovarastoon ladattaessa, jotta tietovaraston käyttäjä tietää, että kyseisiä asiakkaita ei huomioida kyselyissä. Muuten raporteista löytyisi väärää tietoa eikä liiketoiminta näinollen voisi hyödyntää tietoja.

2.4.2 Tietovaraston hyödyntäminen ja raportointi

Tiedon historiatiedon tallentaminen tietovarastoon on vasta ensimmäinen vaihe tietovarastojen hyödyntämisessä. Pelkkä tiedon tallentaminen ei kuitenkaan ole tietovaraston perimmäinen tarkoitus vaikka se onkin erittäin suuri etu. Tallennettua tietoa halutaan kuitenkin hyödyntää, jotta saadaan tietovarastosta kaikki irti. Tiedon hyödyntämiseen on monia eri tapoja. Näistä liiketoiminnalle hyödyllisimmät ovat raportit, erilaiset koontinäytöt ja mittaristot. Näihin tarkoituksiin on olemassa erilaisia valmisohjelmia, sillä tietovaraston käyttöliittymäohjelmasta (DBMS) on hankala saada visuaalisia ja helposti luettavia sisältöjä. Tarvitaan siis omat sovelluksensa, jotka lukevat tietovarastojen sisältämää tietoa, ja joihin on valmiiksi ohjelmoitu tietovarasto-asiantuntijan tekemiä SQL-kyselyitä (Hammergren 2009).

Tietovarastojen datasta suurin osa päätyy raporteille. Raporttien tarkoituksena on visualisoida liiketoiminnalle helposti ymmärrettävää tietoa. Raportointivälineitä löytyy maailmasta paljon ja onkin yrityksen oma asia päättää, mitä niistä juuri heidän kannattaisi hyödyntää. Fiksuinta onkin hoitaa raportteihin liittyvä laskenta ja logiikka tietovarastoissa ja ETL-latauksissa (ei raporteilla). Näin raportin lukeminen käyttäjälle on mahdollisimman jouhevaa, eikä aikaa kulu raportin tietojen lataamiseen. ETL-latausten avulla saadaan datasta muokattua juuri sellaista, kuin raporteilla halutaan nähdä. Tämä nopeuttaa raporttien virkistystä. Useimmissa yrityksissä halutaan, että raporteilla on tuoreimmat tiedot heti aamusta, kun tullaan töihin. Tästä johtuen tietovarastoissa täytyy olla tuorein data heti aamusta. Tämä mahdollistetaan ajastamalla ETL-ajot siten, että ne ovat valmistuneet ennen kuin liiketoiminta aloittaa työt.

2.5 Operatiivinen tietokanta vs tietovarasto

Suurin ero operatiivisen tietokannan ja tietovaraston välillä on, että operatiiviseen tietokantaan luodaan käyttöliittymän avulla tietoa, kun taas tietovarastoon tiedot ladataan toisesta järjestelmästä eräajoilla. Näitä tietoja yhdistellään, historioidaan ja luodaan suurempi kuva tiedosta. Tietovarastossa voi olla useamman sovelluksen tietoa, kun taas operatiivinen tietokanta on suoraan yhteydessä yhteen sovellukseen, josta käyttäjä voi muokata tietokannan sisältöä. Operatiivinen kanta on kooltaan pienempi kuin tietovarasto, joka saattaa usean lähdejärjestelmän ja historiatietojen takia kasvaa erittäin suureksi. (Hovi ym. 2001, 45-48) Guinnes World Recordsin mukaan suurin mitattu tietovarasto on vuonna 2014 ollut kuuden amerikkalaisyrityksen (Sap, Bmmsoft, Hp, Intel, Netapp, Red Hat) yhteinen tietovarasto. Sen koko oli yhteensä 12,1 petatavua eli 12 100 teratavua (GuinnesWorldRecords.com 2014).

Taulukko 2. Operatiivisen tietokannan ja tietovaraston keskeisimmät erot. (Hovi ym. 2001, 46-47)

Toiminto	**Operatiivinen tietokanta**	**Tietovarasto**
Ajantasaisuus	Tärkeää sovelluksen toiminnan kannalta	Ei yleensä tärkeää, kunhan tiedetään milloin uusimmat tiedot ladattu
Käyttö	Paljon pistemäisiä, lyhyitä päivityksiä	Suuri joukko ladataan
Peruskäyttäjien oikeudet	Käyttöliittymän kautta luku- ja muokkausoikeudet	Vain lukuoikeudet, nekin yleensä raporttien kautta

Sarakkeiden tiedot	Saattavat olla vaikeaselkoisia tai koodeina	Eivät saa olla vaikeaselkoisia, vaan selväkielisiä ja aukipurettuja
Taulujen ja sarakkeiden nimet	Saattavat olla vaikeaselkoisia	Eivät saa olla vaikeaselkoisia, vaan selväkielisiä
Taulujen sisältötietojen muuttaminen ja uudet tiedot	Tapahtuu operatiivisen järjestelmän käyttöliittymän kautta tai automatisoidusti	Tapahtuu ETL-latauksilla
Tietojen historiointi	Ei yleensä säilytetä historiatietoja	Historiatietoja säilytetään ennalta määritellyltä aikaväliltä
Yhteys sovelluksiin	Jatkuvassa yhteydessä operatiiviseen järjestelmään	Tiedon lataamisen aikana ETL-latausten kautta yhteys lähdejärjestelmiin. Lukuvaiheessa yhteys raportointisovellukseen.

3 Mallintaminen

Tietokantamallinnuksella tarkoitetaan tietokannan rakennusvaiheessa tehtävää suunnittelua, jossa tietokantakuvaukset muodostetaan. Hyvin suunniteltu on puoliksi tehty. Tietovarastoinnin maailmassa käytetään useimmiten joko tähtimallia (Star schema) tai lumihiutalemallia (Snowflake schema). Nämä mallinnustavat poikkeavat toisistaan silmämääräisesti vain hieman, mutta toiminnallisuuden kannalta niiden välinen ero on kuitenkin oleellinen. (Kimball ym. 2008, 338-339)

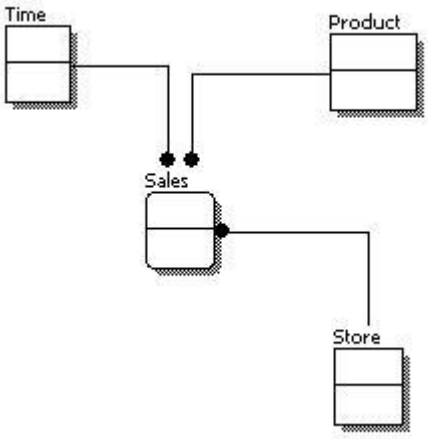
Tietovarastomallinnuksessa puhutaan tietojen normalisoinnista. Normalisoinnilla tarkoitetaan tietokannan tietojen vaiheittaista mallinnusta, jolla tietovaraston attribuutteja jaetaan erillisiin tauluihin. Näin tietoja, jotka voivat esiintyä monella rivillä, ei tarvitsisi kirjoittaa uudestaan. (Kimball ym. 2008, 235-236) Esimerkiksi aikaisemmin mainitut postinumerot: postinumerotaulusta löytyy sarakkeet Postinumero, Kaupunki ja Lääni. Asiakkaan osoitetiedoissa mainitaan vain asiakkaan postinumero. Käyttämällä tätä postinumeroa voidaan kyselyssä liitosten avulla hakea asiakkaan kaupunki ja lääni. Näin säästetään tietovarastoissa hieman tilaa, kun kaupunkia ja lääniä ei tarvitse lisätä asiakkaan omalle riville.

3.1 Tietokantamallinnuksen vaiheet

Tietokantamallinnuksen vaiheita ovat käsitteellinen, looginen ja fyysinen mallintaminen. Näiden vaiheiden avulla saadaan rakennettua tietokannan yhteydet sekä luodaan ensimmäiset osat tietokannan dokumentaatiosta, jota tullaan tarvitsemaan tietokannan ylläpidossa sekä jatkokehityksessä.(1KeyData 2017a)

3.1.1 Käsitteellinen mallintaminen

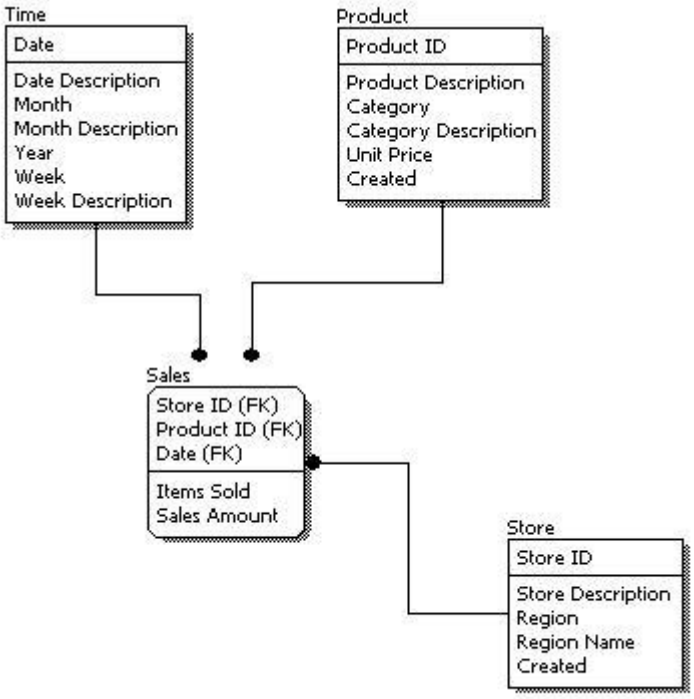
Käsitteellinen mallintaminen on tietokantamallinnuksen suunnittelun ensimmäinen vaihe: tietokannasta tehdään ensimmäinen malli ja kerrotaan tietokannan entiteettien väliset yhteydet ja niiden nimet. Käsitteellinen mallintaminen on erittäin tärkeää tietokannan suunnittelun kannalta, sillä siinä saadaan yleiskuva, miten tietokannassa tietoa tulee käsitellä. Tietovarastoissa käsitteellinen mallinnus saatetaan suorittaa loogisen mallinnuksen yhteydessä. (1KeyData 2017a)



Kuva 3. Esimerkki käsitteellisen mallin luonnoksesta, jossa entiteettien yhteydet on määritelty (1KeyData 2017b)

3.1.2 Looginen mallintaminen

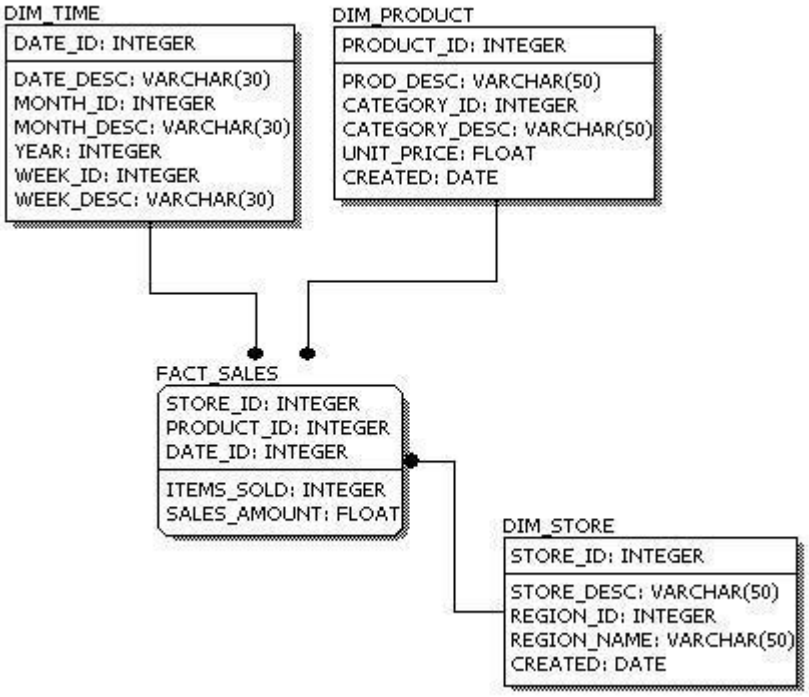
Tietokantamallinnuksen suunnittelun seuraava vaihe on looginen mallintaminen, jossa tietokantakuvaus viedään tarkemmalle tasolle. Loogisessa mallinnuksessa määritellään entiteeteille myös pää- ja viiteavaimet, joilla luodaan entiteettien väliset yhteydet. Loogisessa mallinnuksessa käytetään relaatiotietomallia. (1KeyData 2017)



Kuva 4. Esimerkki loogisen mallin luonnoksesta, jossa avaimet on jo muodostettu (1KeyData 2017c)

3.1.3 Fyysinen mallintaminen

Fyysinen malli on tietokantamallintamisen viimeisen vaiheen tuotos. Tässä vaiheessa tietokanta on jo suunniteltu ja fyysisen mallin avulla tehdään loogisesta suunnitelmasta fyysinen tietokanta. Fyysisen mallinnuksen tuloksena saadaan tehtyä tietokannan luontilauseet. Fyysisen mallintamisen aikana päätetään, minkä tyyppinen tietovarasto otetaan käyttöön. Yleensä valitaan tarpeiden mukaan joko tähtimalli tai lumihiutalemalli. (1KeyData 2017a)



Kuva 5. Esimerkki tietokannan fyysisestä mallista, jossa attribuutit ovat saaneet jo tietotyyppinsä (1KeyData 2017d)

3.2 Normalisointi

Tietovarastojen normalisoinnilla tarkoitetaan tiedon laadun varmistamista. Tietovarastojen tietoa eheytetään jakamalla attribuutteja useampaan tauluun, näin vältetään tauluissa tiedon toistoa. Normalisointeja varten on asetettu tarkat säännöt, joiden pitää toteutua, jotta normalisointi on hyväksyttävästi toteutettu. Relaatiomallinnuksessa normalisointi tapahtuu osana loogista mallintamista. (Virkki 2012, 2-3)

3.2.1 Ensimmäinen normaalimuoto

Ensimmäisessä normaalimuodossa taulun pääavain määrää taulun muut attribuutit. Ensimmäisen normaalimuodon taulu voisi esimerkiksi olla myyntitaulu, jossa olisi kaikkien myyntiin osallistuneiden tiedot: myyjä, asiakas, tuote ja aikatiedot. Taulu olisi epäselvän

näköinen, vaikeasti luettava ja työläs päivittää. Tietovaraston koko kasvaisi jokaisella rivipäivityksellä turhan paljon, eikä tietovaraston taulujen välillä olisi yhteyksiä. (Virkki 2012, 9-12 25)

3.2.2 Toinen normaalimuoto

Toisen normaalimuodon ehtona on, että jokainen attribuutti riippuu koko pääavaimesta. Attribuutin on oltava riippuvainen koko avaimesta. Toisen normaalimuodon tietovarastossa tauluja on jaettu tarkemmiksi ja estetty turhaa tiedon toistoa. Myyntitaulussa onkin avaintiedot tarvittaviin tauluihin, joista saa haettua tarvittavat tiedot. Miksi myyntitaulussa pitäisi siis olla jokaisella rivillä myyjän kaikki tiedot, kun ne voidaan jakaa omaan tauluunsa. Vain myyntiä koskevat oleelliset tiedot lisätään myyntitauluun. (Virkki 2012, 13-16 26)

3.2.3 Kolmas normaalimuoto

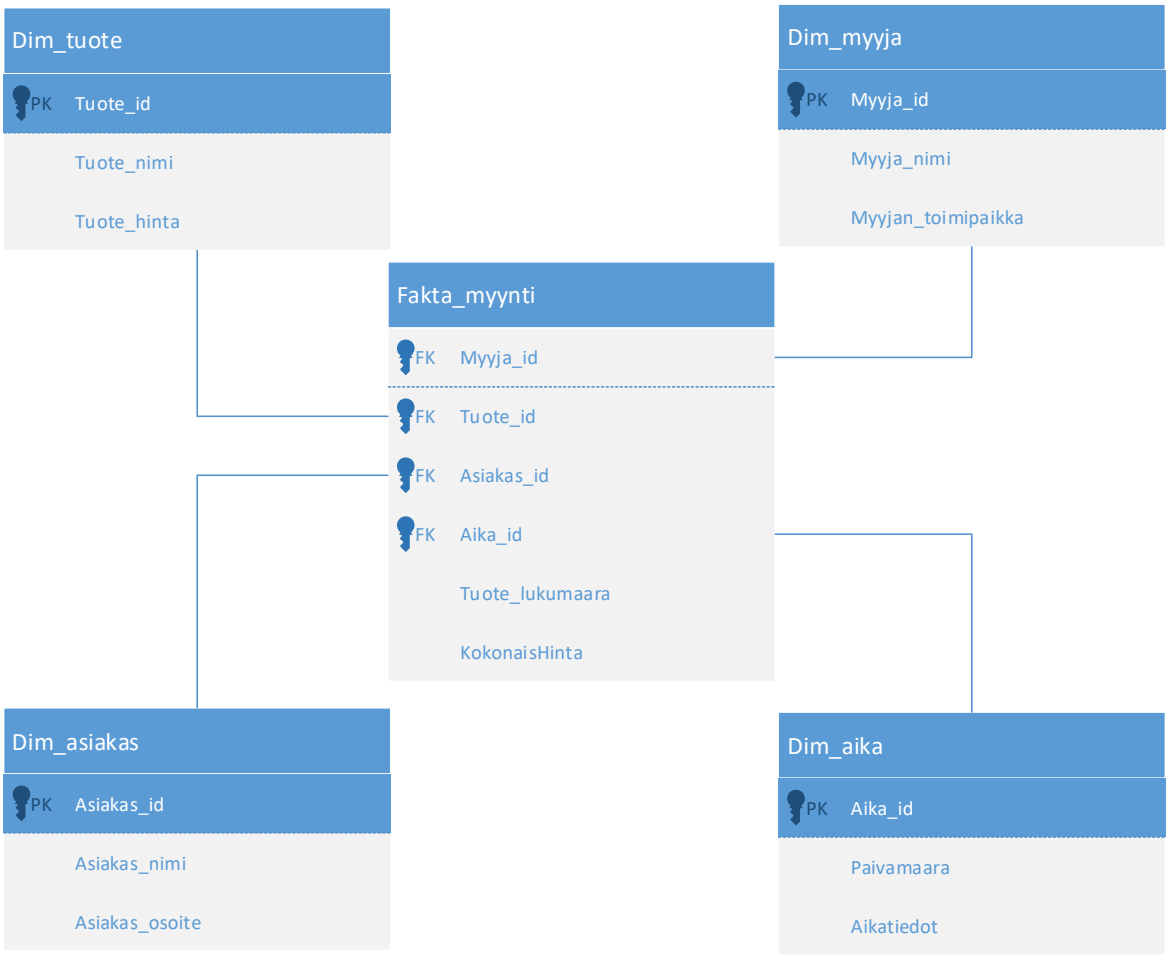
Kolmannessa normaalimuodossa attribuuteilla, jotka eivät ole avaimia, ei saa olla riippuvuuksia muihin tietoihin kuin avaimiin. Kolmannessa normaalimuodossa tiedon toistoa on harvennettu vielä enemmän. Esimerkiksi toisessa normaalimuodossa myyntitaulusta on yhteys myyjätauluun, josta löytyy kaikki myyjän tiedot; nimi, myyjän toimipaikka sekä toimipaikan osoite. Tämäkin saadaan vielä normalisoitua antamalla toimipaikalle oma avaimensa ja tekemällä toimipaikkataulu. Näin toimipaikan kaikki tiedot löytyvät toimipaikkataulusta ja tietovarastossa tauluilla on yhteys avaimen avulla. Näin vältetään myyjätaulussa tiedon toistaminen, sillä saattaahan monella toimipaikalla olla useampi myyjä. (Virkki 2012, 17-20 27)

Pidemmällekin saatetaan normalisoida, mutta relaatiokannoissa tavoitetilana on normalisoida vähintään kolmanteen normaalimuotoon. (Virkki 2012, 22)

3.3 Tähtimalli

Tähtimallilla tarkoitetaan tietovarastoissa käytettävää mallinnustyyppiä. Kyseisessä mallissa tietokanta on normalisoitu toiseen normaalimuotoon. Yleensä relaatiokannoissa normalisoidaan ensin kolmanteen normaalimuotoon, mutta tähtimallissa tietovarasto denormalisoidaan takaisin toiseen normaalimuotoon. Tähtimallinnus on tietovarastoissa yleisimmin käytetty mallinnusmenetelmä sen nopean suorituskyvyn takia. Se on myös lumihiutalemallia helpommin ymmärrettävissä. Tähtimallinnuksella eli dimensiomallinnuksella tietovaraston taulut on jaettu kahteen erityyppiseen tauluun, jotka

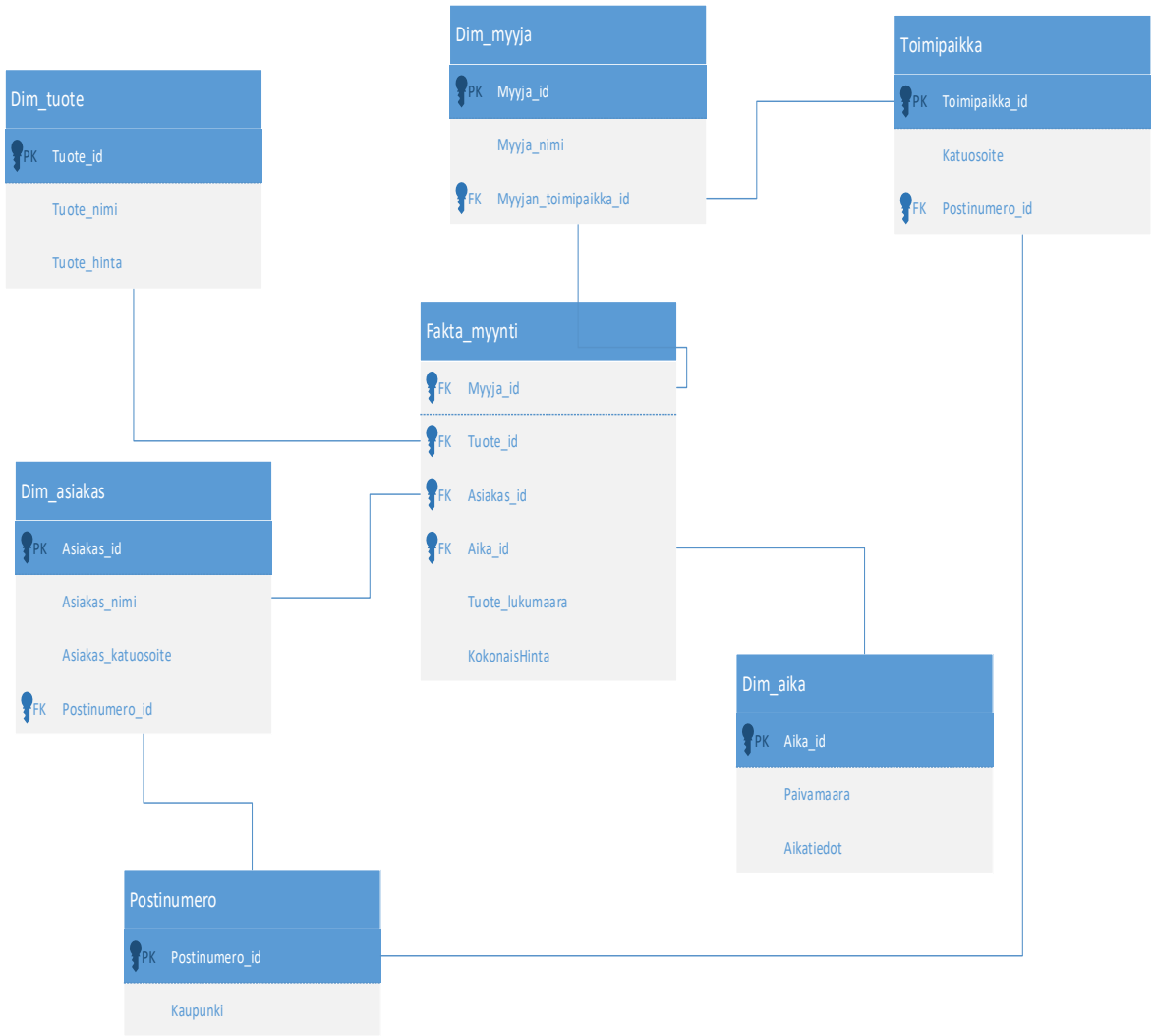
ovat dimensio- ja faktataulut. Faktataulut sisältävät dimensiotauluista yhdistettyjä tietoja sekä tietovaraston laskennalliset tiedot. (Kimball ym. 2008, 235-237)
Esimerkiksi faktatauluna toimisi taulu nimeltä Fakta_myynti. Tauluun liittyisi neljä dimensiotaulua: myyja, asiakas, tuote ja aika. Jokaisella dimensiotaululla on oma yksilöivä avaimensa esimerkiksi myyja_id, asiakas_id, tuote_id ja aika_id. Faktataulun avain muodostuu näistä neljästä avaimesta. Dimensiotauluista löytyy niin sanotut perustiedot, jotka koostavat faktataulun tiedot ja faktatauluista löytyy laskennallisia tietoja, esimerkiksi kokonaismyynti. Näin faktataulun avulla voi luoda esimerkiksi raportin, jossa käydään läpi kaikkien myyjien myyntimäärät ja kuka heistä tuottaa yritykselle eniten rahaa.
Tietovaraston faktatauluja tarkastelemalla voidaan myös saada selville minkälaisia tuotteita tietyt asiakkaat ostavat - näin heihin voisi kohdistaa oikeanlaista mainontaa sekä uusia samantyylisiä tuotteita. Alla on kuva (kuva 7) esimerkkinä tähtimallista.



Kuva 6. Esimerkki tähtimallista

3.4 Lumihiutalemalli

Lumihiutalemallissa tietovarasto on jätetty kolmanteen normaalimuotoon. Kuten tähtimallissa puhutaan edelleen dimensio- ja faktatauluista. Eroavaisuutena tähtimalliin on kuitenkin se, että tietovarasto on jätetty kolmanteen normaalimuotoon ja näin ollen siitä löytyy dimensioille lisädimensiotaulut. (Kimball ym. 2008, 265-267)
Esimerkiksi asiakastaulu: tähtimallissa asiakkaan osoite löytyy kokonaisuudessaan asiakastaulusta. Osoite voidaan kuitenkin hajauttaa postinumeron avulla. Postinumero on yksilöivä tieto ja postinumeroa vastaa aina jokin kaupunki. Toisinpäin tämä ei onnistu, sillä kaupungilla voi olla monta postinumeroa. Samoin myyjätaulussa voidaan myyjän toimipaikka normalisoida omaksi taulukseen. Myyjätauluun jätetään vain toimipaikan yksilöivä tunniste eli id, joka on yhteydessä toimipaikkatauluun. Alla olevassa kuvassa (kuva 8) on esimerkki lumihiutalemallista.



Kuva 7. Esimerkki lumihiutalemallista

4 Takaisinmallinnus

Takaisinmallinnusta on harjoitettu jo ennen tietotekniikan kehitystä: sillä tarkoitetaan mitä tahansa ihmisen tekemän asian purkua, jolla saadaan selvitettyä mitä kaikkea se sisältää ja kuinka se on rakennettu. Takaisinmallinnusta voidaan verrata tieteelliseen tutkimukseen, jossa tutkija yrittää saada selville maailmankaikkeuden ”pohjapiirrosta”. Erona on kuitenkin se, että tieteessä tutkijan selvittämät asiat ovat luontoäidin luomia, kun takaisinmallinnuksessa selvitettävä asia on ihmisen tekemä. Sovelluskehityksessä takaisinmallinnuksella (reverse engineering) tarkoitetaan olemassa olevan teknisen sovelluksen, toiminnallisuuden tai suunnitelman mallin selvittämistä. Takaisinmallinnuksen tarkoituksena on saada selville käytössä olevan sovelluksen puuttuvia ja dokumentoimattomia tietoja, jotka kehityksen aikana on jätetty tekemättä.(Eilam 2005)

Esimerkiksi tilanne, jossa yritys on ulkoistanut tietyn osan toiminnastaan ulkoiselle toimittajalle. Ulkoinen toimittaja on tehnyt asiat oikein ja asiakas on ollut tyytyväinen. Toimittaja on myös kehittänyt vanhoja ja luonut uusia toiminnallisuuksia. Mutta koska heiltä ei ole vaadittu, eivät he välttämättä ole dokumentoineet tekemiään asioita selkeästi tai he ovat jättäneet ne kokonaan dokumentoimatta. Kaikki heidän tarkat tietonsa on ns. ”pään sisällä”, eikä ole nähty tarvetta kirjata tietämiään asioita ylös. Asiakas on sokeasti luottanut toimittajaan, eikä ole vaatinut dokumentteja nähtäväksi. Paljon myöhemmin, kun toimittaja on luonut useita uusia toiminnallisuuksia, ratkaisuja ja kehittänyt paljon uutta, päättää asiakas siirtää itselleen takaisin tämän ulkoistetun toiminnallisuuden. Toimittajalla on kuitenkin velvollisuus tuottaa tarvittavat tiedot kaikesta, mitä heidän toimenkuvaansa on kuulunut ulkoistuksen aikana. Tämän he tekevät, kuitenkin erittäin niukasti, mutta silti sopimuksen puitteissa. Toimittajalla on kuitenkin erittäin paljon sisäistä tietoa, mitä ei välttämättä saada kirjattua dokumentteihin siirron yhteydessä. Tässä tapauksessa asiakkaan on itse otettava selvää näistä asioista. Asiakkaan työntekijät joutuvat turvautumaan takaisinmallinnukseen, jotta he saavat dokumentoitua kaiken, mikä vanhalta toimittajalta oli jäänyt dokumentoimatta. Tämä takaisinmallinnus saattaa koskea ihan mitä vain, mitä toimittaja on tehnyt. Kyseessä voisi olla esimerkiksi monimutkainen koodinpätkä, joka pitäisi olla selkeästi kommentoitu, jotta virheiden selvittely olisi helpompaa tai tietovarasto, jonka mallia ei ole saatavilla tai jonka taulut ja sarakkeet on epäselvästi nimettyjä. Tämänlaiseen tietovarastoon tutustuminen uudelta työntekijältä on aikaa vievää ja haastavaa. Yksi mahdollisuus on piirtää tämä malli itse ja koota siihen kaikki taulujen väliset riippuvuudet. Toinen ja huomattavasti helpompi mahdollisuus on viedä tietovaraston taulut ohjelmaan, joka luo taulujen luontilauseista relaatiomallin, jota tutkimalla on helppo saada kuva koko tietovarastosta.

5 Case: Tietovarastokuvausten takaisinmallinnus

Tämä opinnäytetyö tehdään toimeksiantona Keskinäinen Työeläkevakuutusyhtiö Varma-nimiselle yritykselle. Yhtenä Suomen suurimpana yksityisenä eläkeyhtiönä Varma vastaa yli 870 000 suomalaisen eläketurvasta. Vuonna 2016 Varma maksoi eläkkeitä yli 5,3 miljardia euroa (Varma 2017). Varma on myös yksi Suomen suurimmista yksityisen puolen sijoittajista 42,9 miljardin euron sijoituksilla. Varma omistaa lukuisia liike- ja asuinkiinteistöjä eri kaupungeissa Suomessa ja muualla Euroopassa.

5.1 Nykytilanne

Yhtenä Suomen suurimmista eläkeyhtiöistä Varmalla on suuri vastuu asiakkaidensa tiedoista. Tätä varten heillä on oma asiakastietojen hallintajärjestelmänsä, CRM (Customer Relationship Management). Järjestelmän tarkoituksena on hallita Varman asiakkaiden tietoja ja asiakasyritysten kanssa tehtyjä sopimuksia sekä auttaa Varman henkilöstöä ylläpitämään työkykyjohtamiseen liittyviä asioita. Järjestelmästä ei kuitenkaan saa suoraan raportteja, joilla liiketoiminnan keskijohto pystyisi seuraamaan kokonaistilannetta.

Tähän tarkoitukseen Varmassa on kehitetty erillinen tietovarasto, johon ladataan CRM-järjestelmän tietokannasta asiakkaisiin ja sopimuksiin liittyvät tiedot. Tietovarasto nimeltään CRM DW (Customer Relationship Management Data Warehouse) on ulkoisen toimittajan rakentama ja kehittämä ympäristö. Vuoden 2017 aikana Varma on kuitenkin päättänyt siirtää tietovaraston ja tietovarastoraportoinnin kehittämisen ja ylläpidon omien työntekijöidensä vastuulle. Siirtoprosessin yhteydessä on huomattu puutteita tietovarastokuvauksissa sekä tietovarastojen dokumentaatiossa. Tämänkaltaisten dokumentaatioiden tuottaminen on erittäin tärkeää. Esimerkiksi tilanteessa, jossa uusi työntekijä aloittaa Varmassa tietovarastopuolella, on hänen kannaltaan helpompi saada käsitys tietovaraston sisällöstä, mikäli dokumentaatio on kunnossa. Näin säästetään arvokkaita perehdytystunteja, koska muuten muut työntekijät joutuvat selittämään uudelle työntekijälle yksinkertaisia asioita, kuten mitä tietty taulu sisältää ja mitä sen sarakkeet tarkoittavat.

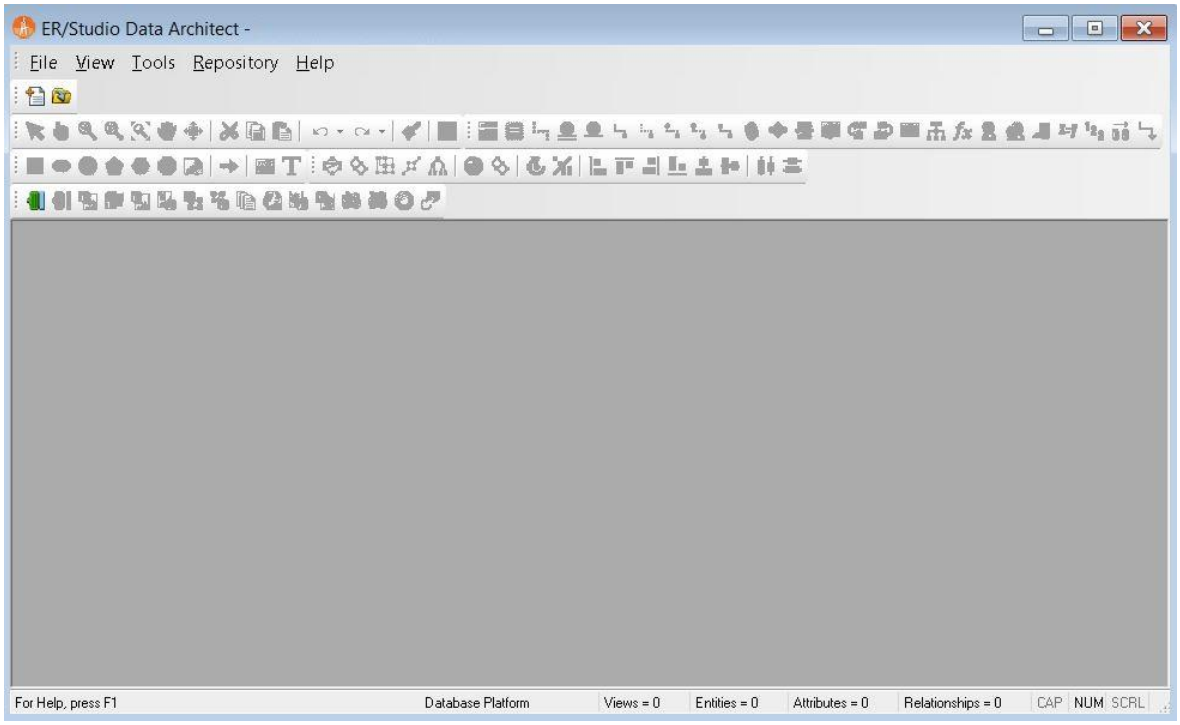
5.2 Tavoite

Tämän toimeksiannon tavoitteena on luoda mallinnustyökalun avulla takaisinmallinnettu tietovarastokuvaus käytössä olevasta tietovarastosta. Ulospäin näkyvänä lopputuloksena mallinnustyökalun avulla saadaan luotua tietovarastosta relaatiomalli, tietovaraston uudet SQL-lauseet kommentoituna sekä erilliselle Varman palvelimelle luotu HTML-sivu, jolla

voidaan tutkia tietovaraston mallia internetselaimella. Mallinnustyökalun avulla syntyneitä uusia SQL-lauseita ei kuitenkaan tulla käyttämään, koska tietovarasto on jo käytössä, mutta on tärkeää tietovaraston kehityksen kannalta, että luontilauseet löytyvät kommentoituna erillisestä dokumentista.

5.3 Käytetyt työkalut

Tavoitetilan saavuttamiseksi työkaluna tulee käyttää mallinnustyökalua, jonka avulla saadaan toteutettua edellä mainitut tavoitteet. Takaisimallinnuksen työkaluksi valikoitui Ideran ER/Studio Data Architect. Työkalun avulla saadaan luettua tietovaraston taulut, joiden avulla työkalu saa luotua vaaditun relaatiomallin. ER/Studio on lisenssituote, jonka käytöstä on maksettava vuosittainen maksu. Varmalla on jo käytössä kyseinen tuote, joten opinnäytetyöstä ei aiheudu lisäkustannuksia Varmalle.



Kuva 8. Yleisnäkymä ER/Studiosta

Tietovaraston tietojen saamiseksi ER/Studioon pitää käyttää ODBC:ta. ODBC (lyhenne sanoista Open Database Connectivity) on yleisessä käytössä oleva standardoitu avoin rajapinta tietokannoille. Varmalla se on käytössä, jotta tietyt ohjelmat saavat helposti luettua tietokantojen taulujen rakenteita.

Toimeksiannon onnistumisen kannalta on myös tärkeä päästä näkemään, miten CRM järjestelmästä tietovarastoon tulleita tietoja on muokattu ja ladattu. Tähän tarkoitukseen pitää käyttää Informaticaa, joka on Varman käytössä oleva ETL ohjelma. Ohjelman

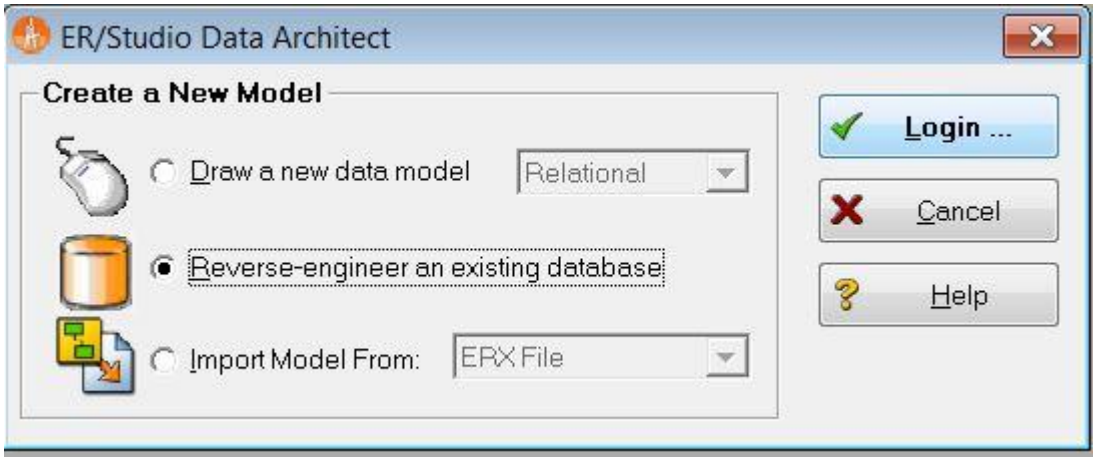
sisältämiä mappingeja tutkimalla saadaan jokaisen tietovaraston taulun sarakkeiden tiedon lähde ja muutos selvitettyä. Mappingit ovat Informaticalla tehtäviä latauskarttoja, joilla määritellään mistä lähteestä tietoa ladataan, miten tietoa muokataan ja minne muokattu tieto ladataan.

Viimeisenä välineenä tarvitaan Varman CRM-järjestelmää ja sen tietokannan tietokuvausta. Tietokannassa olevat tiedot eivät ole aivan yksiselitteisiä, joten on tärkeää tietää, mitä käyttöliittymän tietoa tietokannan kentät tarkoittavat.

5.4 Tavoitteiden saavuttaminen ja työn toteutus

5.4.1 Alkuvalmistelut ja takaisinmallinnus

Työn suorittamisen kannalta aivan ensimmäiseksi on tuotava tietovaraston tiedot ER/Studio-ohjelmaan. Ohjelmaan on sisäänrakennettu takaisinmallinnusta varten oma ohjattu toiminto, johon voidaan valita halutut tietovaraston objektit.



Kuva 9. Valintaikkuna ER/Studiossa uuden työn aloittamiseksi

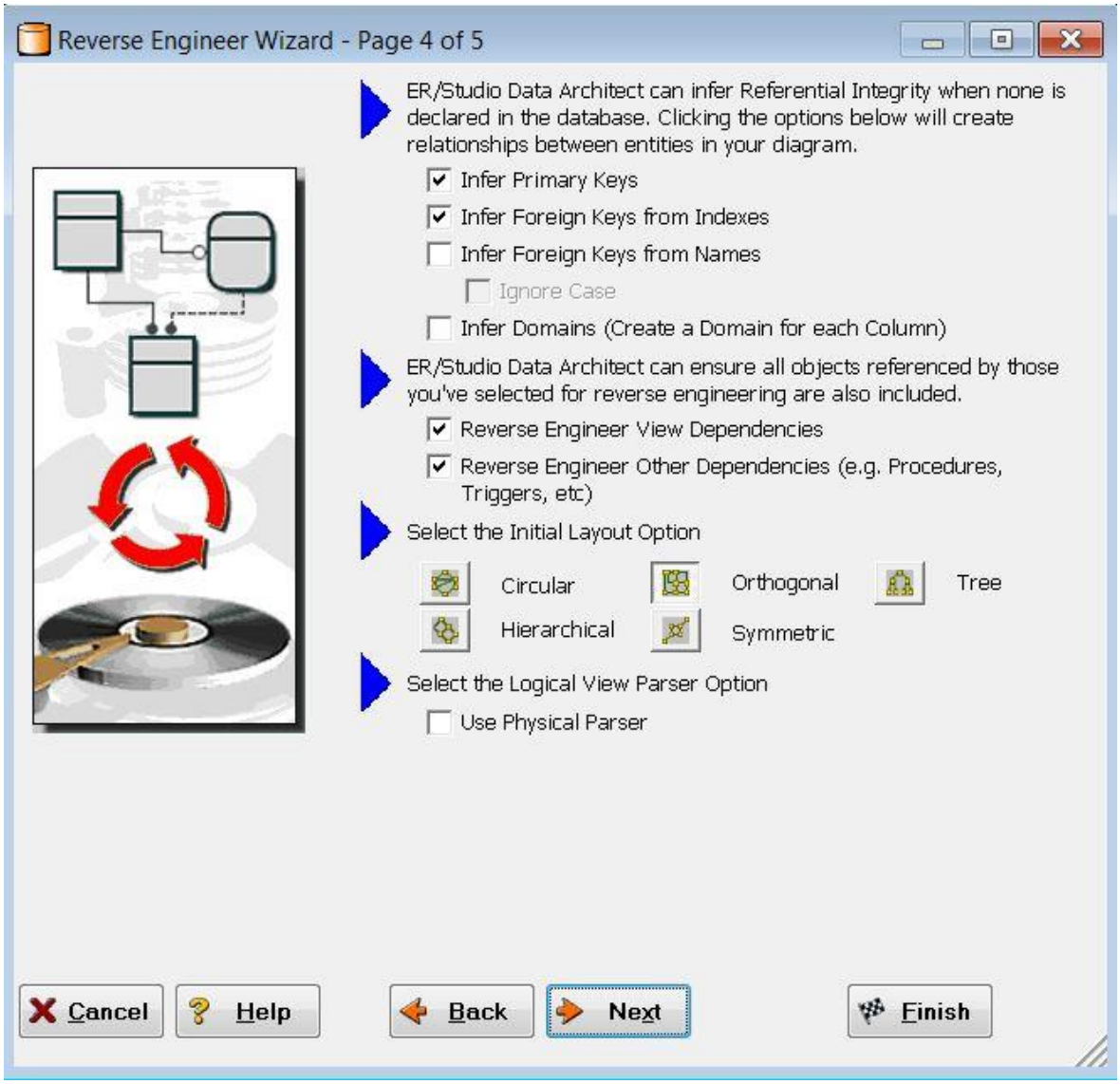
Kuvassa 9 valitaan, miten uutta tietokantakuvausta halutaan aloittaa työstämään. Tässä työssä valitsemme vaihtoehdon kaksi: ’Reverse-engineer an existing database’ eli ’takaisinmallinna olemassa oleva tietokanta’. CRM DW on jo olemassa, joten olisi turhaa työtä aloittaa tyhjältä pöydältä (ensimmäinen vaihtoehto). Kolmannessa vaihtoehdossa voidaan tuoda muulla tavalla tuotettua tietokantakuvausta, esimerkiksi kuvassa näkyvä ERX File on Erwin-nimisellä mallinnustyökalulla tuotetun tietokantamallin tiedostopääte.

Seuraavassa vaiheessa valitaan tapa, jolla otetaan yhteys tietokantaan. Kappaleessa 5.3 esitelty ODBC on tässä kohtaa paras vaihtoehto, koska näin saadaan helpoiten rakennettua yhteys tietovarastoon. ODBC:n toimimista varten tarvitaan tunnukset, joilla

rajapinta saa yhteyden tietovarastoon. Käytän tässä omia tunnuksiani, joilla itse kirjaudun tietovarastoon.

Kuva 10. Lista, josta voidaan valita mitä kaikkea tietokannasta halutaan ER/Studioon tuoda

Seuraavaksi valitaan, mitä kaikkea halutaan tuoda tietokannasta ER/Studioon. Kuvassa 10 on listaus mahdollisista valittavista asioista. Tähän toimeksiantoon kuuluvat kentät ovat User Tables eli käyttäjän luomat taulut, sekä User Views eli käyttäjän luomat näkymät. System Tables ja Views jätetään pois, sillä tietovarastokuvauksiin ei tarvita selvitystä, mitä järjestelmän luomia tauluja ja näkymiä tietovarastosta löytyy. Synonyymit jätetään myös pois, koska niitä ei tästä tietovarastosta löydy. Tämän jälkeen ER/Studio hakee kaikki käyttäjän luomat taulut ja näkymät. Näistä valitaan ne jotka halutaan tietovarastomalliin lisätä. Tässä tapauksessa moni taulu tippui pois, sillä niillä ei ollut tekemistä tietovaraston kannalta tai ne olivat kopioituja tauluja. Tauluja on saatettu kopioida tietovaraston sisällä: esimerkiksi tilanteessa, jossa tauluun tehdään joku muutos. Vanha taulu halutaan säilyttää jonkin aikaa, sillä mikäli myöhemmin havaitaan jotain ongelmia, voidaan aina palata aiemmin oikein toimivaan versioon. Samasta tietokannasta löytyy myös toisen skeeman tauluja, jotka eivät myöskään kuulu CRM DW:hen, joten niitäkään ei oteta mukaan.

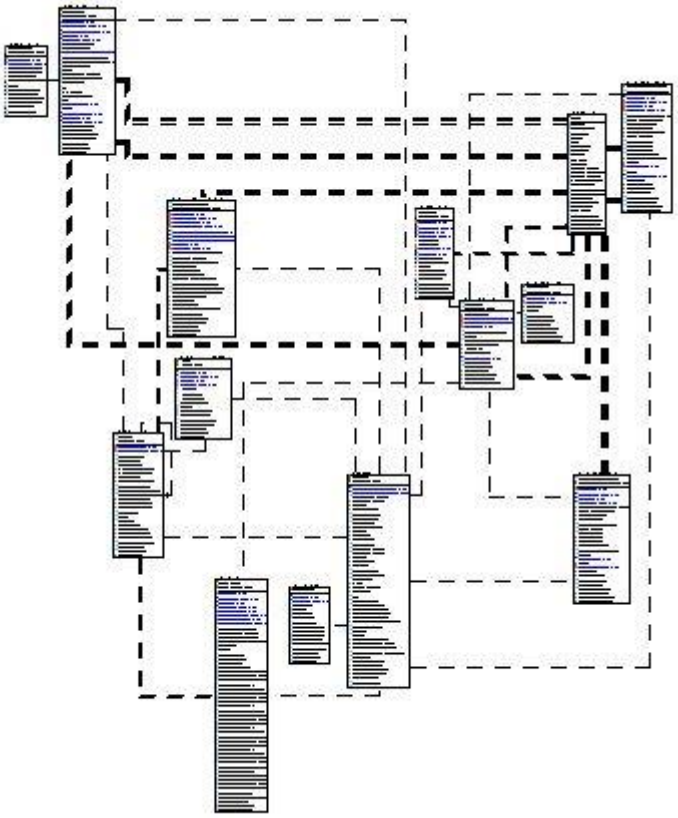


Kuva 11. ER/Studion takaisinmallinnuksen ikkuna, jossa valitaan mallinnukseen liittyviä asioita sekä pohjapiirroksen malli. Pohjapiirroksen voi myöhemmin vaihtaa muuhun vaihtoehtoon

Nyt on vuorossa tietovaraston taulujen pääavaimien ja viiteavaimien yhteydet. Valitsin, että pääavaimet tuodaan ER/Studioon ja viiteavaimet indekseistä. Mikäli viiteavaimet tuodaan nimien avulla, saattaa joitain taulujen välisiä yhteyksiä jäädä luomatta, jos viiteavain ei ole samanniminen kuin pääavain. Indekseihin on kuitenkin liitetty kaikkien taulujen yhteydet, jotta tietovaraston lukunopeus olisi optimaalinen. Tällä sivulla valitsin myös, että takaisinmallintaan lisätään näkymien ja muiden tietokantaosien riippuvuuksia. Tämä tehdään siksi, että saadaan kaikki oleellinen tietomalliin, eikä mitään tärkeitä komponentteja jätetä pois. Sivulla valitaan myös minkälaisen pohjapiirroksen malli ER/Studiossa saa, tämän saa kuitenkin muutettua jälkikäteen, joten tässä vaiheessa sen valitseminen ei ole olennaisessa osassa takaisinmallinnusta.

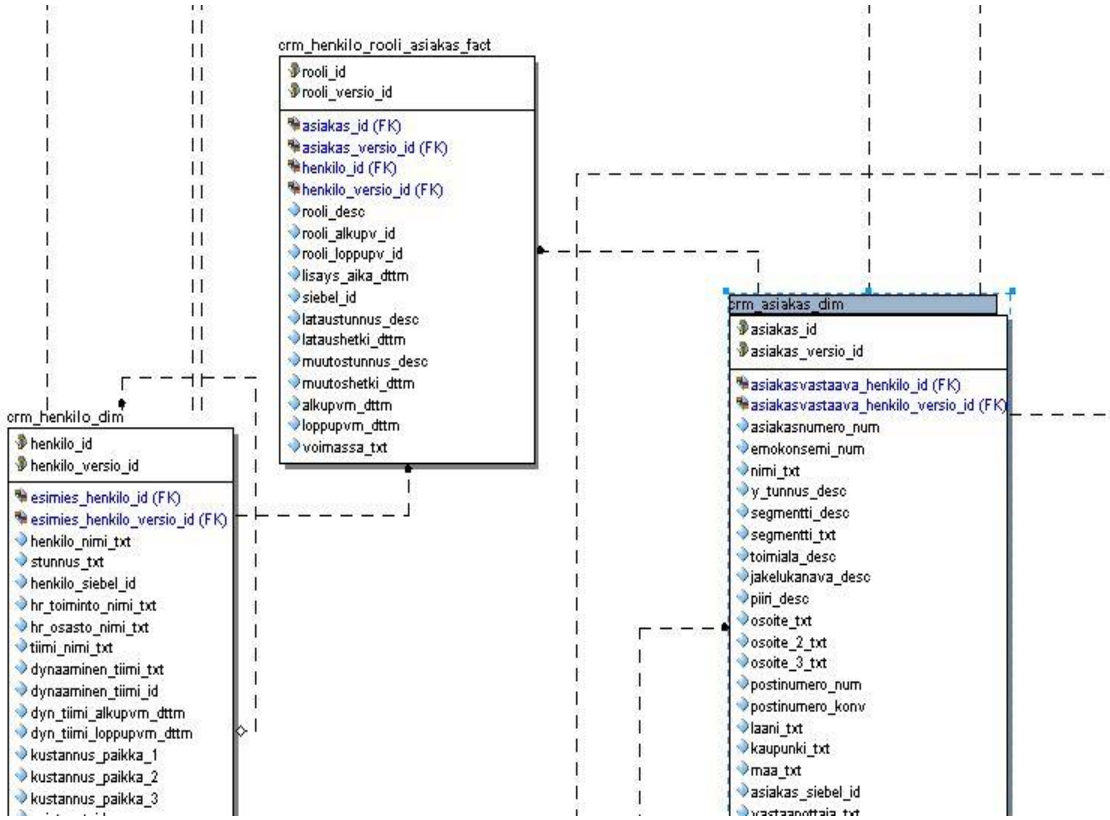
Viimeisessä vaiheessa voidaan tarkastella objekteja, jotka tuodaan malliin. Nyt valitaan myös minkä tyyppinen fyysinen malli on kyseessä. Tähän valitsin dimensionaalisen mallin relaatiomallin sijasta, sillä kyseinen tietovarasto on dimensionaalinen. ER/Studiolle kerrotaan myös, että tämän tietokannan tietokantaympäristö on Microsoftin SQL Server 2012. Lopuksi voidaan vielä tallentaa kaikki täytetyt tiedot, mikäli tietokantamalli pitää luoda uudestaan.

Kaikkien ER/Studion kysymien asetustietojen jälkeen voidaan käynnistää takaisinmallinnus, joka ei kestä kauaa. Lopulta olemme saaneet tarkan mallin tietovaraston tietojen perusteella ja voimme alkaa tutkimaan mallia.



Kuva 12. Ulospäin zoomattu kuva CRM DW:stä, josta näkee taulujen määrän ja niiden väliset yhteydet

Kuvassa 12 näemme yleiskuvan tietovarastosta. Tämän kuvan tarkoituksena on antaa lukijalle kuva siitä, millaiselta tietovaraston mallin tulee näyttää. Kuvasta ei näe tarkkoja taulujen tietoa, sillä tietomalli on iso. Kuvassa 13 on tarkennettu osaan tietovarastosta, jotta nähtäisiin myös taulun nimi ja sarakkeet sekä niiden yhteys toiseen tauluun.

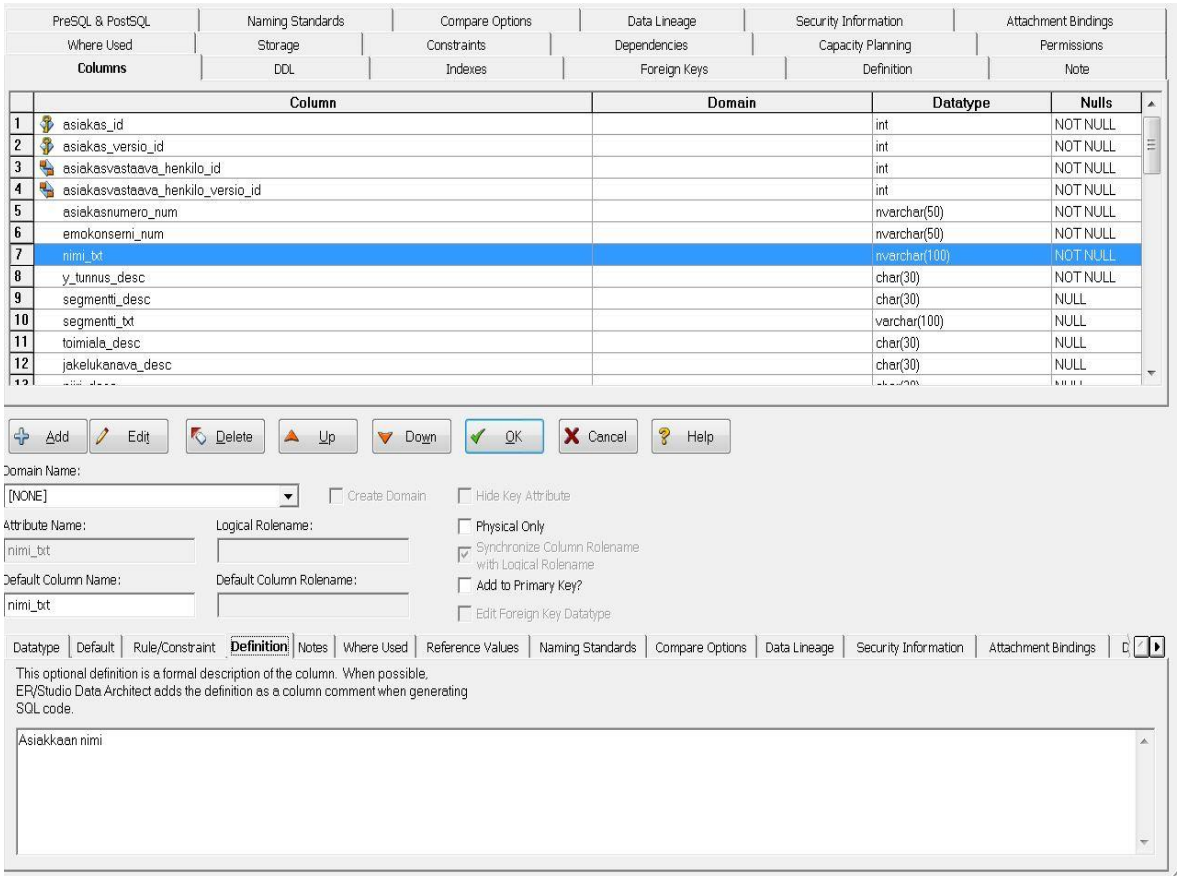


Kuva 13. Henkilo_dim- ja asiakas_dim-taulut, sekä niiden välinen yhteys faktatauluun henkilo_rooli_asiakas_fact

5.4.2 Taulujen ja sarakkeiden tietosisällön selvitys

Toimeksiannon seuraavassa vaiheessa pitää ottaa selvää, mitä taulut ja sen sisältämät sarakkeet tarkoittavat. ER/Studiossa tämän selvityksen tekemiseen auttaa definition (määritelmä) -välilehti. Välilehteen kirjoitettu tieto tulee näkymään SQL-luontilauseissa kommenttina sekä mallin HTML-sivulla. Tauluilla on siis oma määritelmävälilehti ja taulun jokaisella sarakkeella omansa.

Taulujen määritelmät on helppo päätellä taulun nimen perusteella: esimerkiksi crm_asiakas_dim-taulun määritelmä on ”Dimensiotaulu, joka sisältää Varman asiakkaiden tietoja.”. Joidenkin taulujen kanssa oli haasteita: esimerkiksi taulun crm_asialuokittelu nimestä voidaan päätellä, että taulu sisältää asialuokittelu tietoja, mutta ei kuitenkaan voida tietää, mihin kyseiset asialuokittelut liittyvät. Kuitenkin, koska taulu sisältää viiteavaimena sarakkeen aktiviteetti_id, saadaan tietää, että taulu sisältää aktiviteettien asialuokittelutietoja.



Kuva 14. Esimerkkikuva crm_asiakas_dim-taulusta ja sarakkeesta nimi_txt, johon on tehty määritelmä

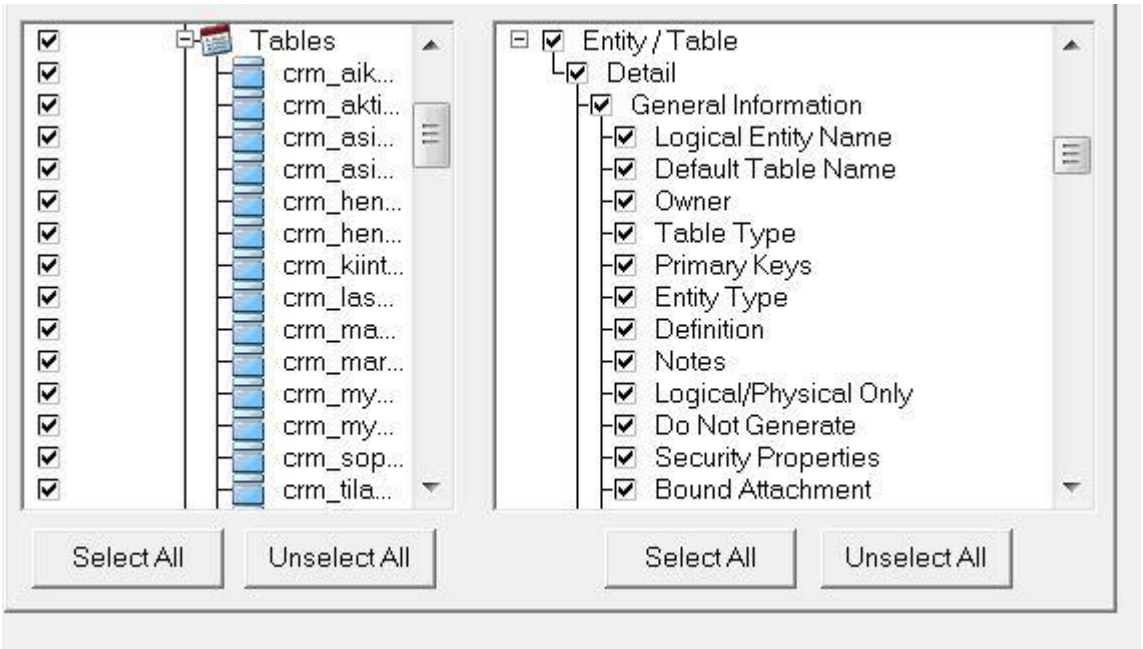
Kuvassa 14 on esitetty käyttöliittymää sekä tapa, miten määritelmä on lisätty sarakkeeseen. Samassa käyttöliittymässä pystytään myös lisäämään uusia rivejä tauluun. Datatype-välilehdeltä voidaan valita uudelle sarakkeelle haluttu tietotyyppi. Kuvan ylhäällä olevista välilehdistä päästään muokkaamaan taulun tietoja. Välilehdeltä DDL (data definition language) saadaan taulua vastaava luontilause, joka sisältää myös näiden sarakkeiden määritelmät erillisinä kommentteina.

Joissain tapauksissa määritelmän saaminen ei ole niin yksinkertaista kuin kuvan 14 esimerkissä. Esimerkiksi asiakas taulussa oleva sarake tyyppi sisältää arvoja ’P’, ’B’ ja tyhjä. Vaikka voidaan päätellä, että kyseessä on asiakastyyppi, tietojen P ja B arvot eivät avaudu välttämättä heti tietovaraston tutkijalle, joten on hyvä, että tietovarastokuvauksesta löytyvät myös selitykset näille kirjainarvoille. Tutkimalla CRM-käyttöliittymää havaitaan, että ’asiakastyyppi’-kenttä voi käyttöliittymässä saada arvot henkilöasiakas tai yritysasiakas. Informaticaa tutkimalla voidaan havaita, että P on lyhenne sanasta Person eli henkilö ja B on lyhenne sanasta Business eli yritys.

Samaa toimintoperiaatetta käytettiin kaikkiin tietovaraston tauluihin ja näkymiin (yhteensä 35 kappaletta) sekä taulujen sisältämiin sarakkeisiin (tietovarastossa on yli 550 kappaletta). Kaikki sarakkeiden määritelmät varmistettiin lukemalla tietokantaa, tutkimalla ETL-latauksia tai loppukädessä etsimällä CRM-järjestelmän käyttöliittymältä tietoa vastaava lähdekenttä. Näin varmistettiin määritelmien oikeellisuus.

5.4.3 Tietovarastomallin kuvaukset HTML-raportin

Toimeksiannon viimeisessä vaiheessa tuotetaan ER/Studiolla luodut kuvaukset ja määritelmät HTML-pohjaiseksi raportiksi, johon päästään helposti selaimella käsiksi. Tämä vaihe toteutetaan myös ER/Studiolla. Ohjelmaan on sisäänrakennettu työkalu, jolla on mahdollista luoda HTML-raportteja. Haluamme käyttää juuri HTML-raporttia, jotta muidenkin on helppo lukea mallia selaimella. Ohjatun toiminnon avulla valitaan, mitä kaikkea raporttiin halutaan tietomallista lisätä. Kuvan 15 kohdassa (alla) valitsin, että kaikki mallin taulut, näkymät ja taulujen väliset yhteydet kuvataan raporttiin. Tauluista valitsin myös kaikki mallin sisältämät tiedot taulun nimestä tietoturvatietoihin.



Kuva 15. ER/Studion raportinluonti-työkalulla voidaan valita vain ne tietyt taulut, jotka halutaan mallista HTML-raportille

Työkalulla luodaan lopuksi mallille omistustiedot, joista nähdään, kuka mallin on luonut, mallin nimi sekä yritys- ja tekijänoikeustiedot. Nämä ovat välttämättömiä tietoja, jotta tiedetään kenen tekemä malli, minä vuonna malli on luotu ja kenelle kuuluvat mallin tekijänoikeudet. HTML-raportilla nämä tiedot näkyvät etusivulla, näin nähdään heti, mistä mallista on kyse.

ER/Studion HTML-raportti sisältää saman tietomallin kuin ohjelma itsessään. Raportilta pääsee tutkimaan samoja tietoja, joita toimeksiannon aikaisemmassa vaiheessa lisättiin sekä taulujen luontia varten olevat SQL-lauseita (kuva 16). Mitään erillistä dokumentaatiota ei siis tietokannan mallista tarvitse enää rakentaa, sillä mallin ylläpitäminen toimii ER/Studiolla. Näin saadaan myös keskitetysti pidettyä tietokantaan liittyvät tiedot yhdessä ja samassa paikassa. Tietovaraston yleisdokumenttiin tai tekniseen dokumentaatioon voidaan esimerkiksi lisätä linkki ER/Studion HTML-malliin.

ColumnName	Domain	Datatype	NULL	Definition
asiakasnumero_num		nvarchar(50)	NO	Asiakkaan tunnistenumero
emokonserni_num		nvarchar(50)	NO	Asiakkaan emokonsernin tunnistenumero
nimi_txt		nvarchar(100)	NO	Asiakkaan nimi

Kuva 16. Ote HTML-raportilta asiakastaulun sarakkeiden tietokuvauksista

Kuten jo aikaisemmin opinnäytetyössä olen kertonut, ER/Studio on lisenssituote eli sen käyttö vaatii käyttäjältä maksua. HTML-raporttien lukeminen on kuitenkin täysin ilmaista, ne voidaan siis esimerkiksi laittaa johonkin yleiseen hakemistoon yrityksen muun henkilökunnan nähtäville tai sijoittaa jollekin palvelimelle, johon voidaan päästä käsiksi suoraan selaimen osoiterivillä. Mitään muutoksia ei pysty tekemään selaimen kautta, vaan kaikki muutokset pitää tehdä ER/Studio-ohjelmassa ja luoda muutoksen jälkeen raportti uudestaan. Varmalla on käytössä ulkoisen toimittajan luoma scriptitiedosto, jolla haetaan joka päivä uusin versio tietomalleista. Näin saadaan aina uusin tietomalli kaikkien käyttäjien nähtäville.

5.5 Yhteenveto

Takaisinmallinnuksessa haastellisin osa oli löytää tiedot sarakkeisiin, jotka eivät olleet yksiselitteisiä tai sarakkeen arvo ei antanut tutkijalle järkevää infoa (esimerkiksi asiakastaulun sarake ’tyyppi’ kappaleessa 5.4.2). Paljon aikaa vei myös tietojen tarkistus: sarakkeen nimestä voitiin olettaa paljon, täytyi se kuitenkin tarkistaa, jotta voitiin olla varmoja tietojen oikeellisuudesta. Jatkon kannalta on kuitenkin huomattavasti parempi, että kaikki tiedot ovat oikein: miten voi luottaa tietomalliin, jos tiedot ovat väärin? Sama näkökulma pätee ihan suoraan tietovarastoon; jos tietoja latautuu väärin, eivät ne näy raporteillakaan oikein. Liiketoiminnan täytyy voida luottaa tietovarastoon ja sen käyttöön, muuten tietovarastosta tulisi hyödytön hukkainvestointi. On siis erittäin tärkeää huolehtia

niin tietovarastosta kuin siihen liittyvästä dokumentaatiosta. Jatkossa kun tietovarastoa kehitetään eteenpäin, voidaan käyttää ER/Studion tietomallia ja päivittää sinne uudet taulut ja sarakkeet.

Suositeltavaa olisi, että jokaisesta yrityksen tietovarastosta tehtäisiin samankaltainen ER/Studion tietomalli, näin saataisiin keskitettyä kaikkien tietovarastojen tietomallidokumentaatiot samaan paikkaan. HTML-raporttien avulla tämä on mahdollista. Erityisesti uusissa tietovarastoprojekteissa on suositeltavaa luoda ER/Studiolla tietovaraston malli, jotta dokumentaatiot saadaan alusta lähtien tehtyä kunnolla.

6 Työtuloksen pohdinta

Opinnäytetyön tavoitteena oli luoda olemassa olevasta tietovarastosta takaisinmallinnettu tietomalli, joka sisältää kaikkien taulujen ja sarakkeiden tarkat tiedot. Mielestäni projekti onnistui tavoitteessaan täydellisesti, mitään ei jäänyt tietomallista puuttumaan ja tietomalli voidaan ottaa välittömästi käyttöön. Projektin aikana tietovarastoa myös kehitettiin jatkuvasti, joten oli tärkeää olla tietoinen kehityksestä, jotta tietovaraston uusimmat tiedot saatiin otettua tietomalliin mukaan. Niiden puuttuminen olisi aiheuttanut luottamuspulaa tekijää kohtaan.

Oman oppimisen kannalta olen työhön erittäin tyytyväinen. Opin erittäin paljon dokumentaation hyödyistä ja sen puuttumisen haitoista, sekä siitä miten työlästä ja aikaa vievää jo tehdyn asian tutkiminen on. Opin myös tietokantojen toiminnasta paljon sekä miten tietovarastot toimivat, ja miten ne eroavat normaaleista operatiivisista tietokannoista. Olisin halunnut oppia myös tietovaraston käytöstä liiketoiminnan puolelta. Se ei kuitenkaan kuulunut tämän opinnäytetyön suunnitelmaan ja olisi luultavasti kasvattanut työmäärän kaksinkertaiseksi.

Opinnäytetyössä oli myös aikataulullisia haasteita, sillä takaisinmallinnuksen vaatima tiedon etsiminen vei odotettua enemmän aikaa. Tämä oli kuitenkin loppujen lopuksi hyvä asia, koska se avasi silmäni dokumentaation tarpeelle. Uskonkin olevani tulevaisuudessa huolellisempi, en vain silloin, kun itse laadin dokumentaatiota, vaan myös kun odotan jonkun muun tekevän dokumentaatiota. Dokumentaation tekeminen sovelluskehityksessä saattaa kehittäjästä tuntua turhan raskaalta, mutta loppujen lopuksi kehittäjä kiittää itseään ja dokumentaatiota ongelmien ilmaantuessa tai jatkokehityksessä.

Lähteet

1KeyData. 2017a. Data Warehousing Concepts. Luettavissa: http://www.1keydata.com/datawarehousing/concepts.html. Luettu. 20.4.2017.

1KeyData. 2017b. Conceptual data model. Luettavissa: http://www.1keydata.com/datawarehousing/conceptual-data-model.jpg. Luettu: 20.4.2017.

1KeyData. 2017c. Logical data model Luettavissa: http://www.1keydata.com/datawarehousing/logical-data-model.jpg. Luettu: 20.4.2017.

1KeyData. 2017d. Physical data model. Luettavissa: http://www.1keydata.com/datawarehousing/physical-data-model.jpg. Luettu: 20.4.2017.

Chamberlin, D. & Boyce, R. 1974. Sequel: A structured English query language. Luettavissa: http://www.almaden.ibm.com/cs/people/chamberlin/sequel-1974.pdf. Luettu: 15.4.2017.

Chapple, M. 2016. SQL Fundamentals. Luettavissa: https://www.thoughtco.com/sql-fundamentals-1019780. Luettu: 15.4.2017.

Eilam, E. 2005. Reversing: Secrets of Reverse Engineering. 1. painos. Wiley Publishing Oy. Indianapolis, US.

GeekInterview 2007. What is Operational Database. Luettavissa: http://www.learn.geekinterview.com/data-warehouse/dw-basics/what-is-operational-database.html. Luettu: 11.4.2017.

Guinnes World Records. 2014. Largest data warehouse. Luettavissa: http://www.guinnessworldrecords.com/world-records/largest-data-warehouse. Luettu: 11.4.2017.

Hammergren, T. 2009. Querying and reporting tools for data warehousing. Luettavissa: http://www.dummies.com/programming/big-data/engineering/querying-and-reporting-tools-for-data-warehousing/. Luettu: 13.4.2017.

Hovi, A., Ylinen, J. & Koistinen, H. 2001. Tietovarastot liiketoiminnan tukena. 1. painos. Satku. Jyväskylä. S. 27-37, 45-48, 76-81.

Kimball, R., Ross, M., Thornthwaite, W., Mundy, J. & Becker, B. 2008. The Data Warehouse Lifecycle Toolkit. 2. painos. Wiley Publishing Oy. Indianapolis, US. S. 127-128, 235-237, 265-267, 338-339.

Merriam-Webster. Definition of Database. Luettavissa: https://www.merriam-webster.com/dictionary/database. Luettu: 10.4.2017.

Oppel, A. 2010. Databases demystified – a self teaching guide. 2. Painos. The McGraw-Hill Companies. San Francisco, CA, US. S. 13-14

Serra, T. 2013. Why You Need a Data Warehouse. Luettavissa: http://www.jamesserra.com/archive/2013/07/why-you-need-a-data-warehouse/. Luettu: 13.4.2017.

Tietokaira. Tietovarastointi. Luettavissa: http://www.tietokaira.fi/tuotteet-ja-palvelut/tietovarastointi. Luettu: 10.4.2017.

Troels, A. 2011. Comparison of different SQL implementations. Luettavissa: http://troels.arvin.dk/db/rdbms/. Luettu: 28.4.2017

Varma. 2017. Vuosikertomus. Luettavissa: https://www.varma.fi/muut/yhtiotietoa/vuosikertomus/. Luettu 21.4.2017.

Wikipedia: Datawarehouse. Luettavissa: (https://en.wikipedia.org/wiki/File:Datawarehouse.png). Luettu: 15.4.2017.

Virkki, O. Tiedonhallinta ja tietokannat. Normalisointi. Haaga-Helia 2012. Luettavissa: http://myy.haaga-helia.fi/~ict1tn005/materiaalit/ict05_S3_norm.pdf Luettu 15.4.2017. S. 2-3, 9-20, 22, 25-27.